Desenvolvimento de Sistema Web para Gerenciamento de Bancas Avaliadoras de Trabalho de Conclusão de Curso

Denis Macias Veiga[1], Felipe José Dellê[1]

[1]Tecnologia em Análise e Desenvolvimento de Sistemas - Faculdade Guairacá - Caixa Postal 85.010-100 - Guarapuava - PR - Brasil

deniscode0@gmail.com, felipedelle@yahoo.com.br

Abstract. *This article aims to present the development of a web system to manage the examination boards of undergraduate courses Guairacá College. The system was developed using PHP programming language, front-end framework Bootstrap and SGBD MySQL. The project was based on the MVC software architecture model and was used the prototyping as approach of software engineering. The system provided greater productivity, control and safety on management of examination boards of the institution.*

Resumo. *Este artigo tem como objetivo apresentar o desenvolvimento de um sistema web para gerenciar as bancas avaliadoras dos cursos de graduação da Faculdade Guairacá. O sistema foi desenvolvido utilizando a linguagem de programação PHP, framework front-end Bootstrap e o SGBD MySQL. O projeto foi baseado no modelo de arquitetura de software MVC e a abordagem da Engenharia de Software utilizada foi a prototipação. O sistema propiciou maior produtividade, controle e segurança no gerenciamento de bancas avaliadoras da instituição.*

1. Introdução

A utilização de sistemas de informação por qualquer tipo de organização é quase imprescindível devido ao grande volume de dados e informações que uma entidade organizacional pode gerar. Em se tratando de instituições de ensino, a necessidade de gerar processos ágeis e seguros aumenta exponencialmente, pois garantem a produtividade, eficiência e segurança do tráfego de dados e informações na instituição. Essa necessidade foi o grande motivador da realização deste trabalho.

A Faculdade Guairacá não possui um sistema informatizado para gerenciamento das bancas avaliadoras dos Trabalhos de Conclusão de Curso. Todo o controle de bancas da instituição é realizado manualmente, utilizando editores de textos, planilhas eletrônicas ou até mesmo agendas manuscritas. Esse tipo de controle além de não ser ágil, é impreciso, pois para o lançamento de uma banca deve-se verificar se os integrantes estão disponíveis no dia e horário desejado para o cadastro da nova banca, e ainda verificar se a sala não está sendo ocupada no dia e horário. Toda essa checagem é manual, lenta e suscetível a erros.

Para o desenvolvimento do sistema foi utilizado a linguagem de programação PHP, linguagem de marcação HTML e a estilização com a linguagem CSS. Foram utilizadas as seguintes ferramentas para o desenvolvimento do sistema: *Notepad++* para a codificação, *MySql Workbench 6.1 CE* para a modelagem do banco de dados, e os diagramas foram criados fazendo uso do software *Astah Community 6.9.0*. Para garantir a responsividade do sistema foi utilizado o *framework Bootstrap*. Como apoio à

formatação e validação de campos de entrada de dados, fez-se uso da biblioteca *Javascript jQuery* e do *plugin jqBootstrapValidation*.

A linguagem de modelagem aplicada neste trabalho foi a *Unified Modeling Language* (UML), fazendo uso especificamente dos diagramas de classes, caso de uso e de sequência. Em relação ao padrão de projeto (*Design Pattern*), foi utilizado o modelo *Model, View, Controller* (MVC).

A objetivo deste projeto é desenvolver um sistema *Web* que gerencie as bancas avaliadoras de Trabalhos de Conclusão de Curso da Faculdade Guairacá – Guarapuava - PR, tornando o controle mais rápido e seguro. O usuário não precisará fazer nenhum tipo de verificação, pois o sistema realiza todas as validações necessárias para que não ocorram inserções ou alterações indevidas. Com isso, o processo de gerenciamento das bancas torna-se mais eficiente.

2. Fundamentação Teórica

Os Sistemas de Informação (SI) apresentam como desafio garantir a agilidade da informação e sua qualidade. Antes de tudo, a informação deve ser realmente útil e atender as necessidades dos seus usuários ou gestores. Dentro deste contexto, caracteriza-se Tecnologia da Informação (TI) como instrumento essencial para a continuidade e ascensão das organizações. Para [Oliveira 2006] a tecnologia da informação é imprescindível não somente pelo recurso tecnológico, mas principalmente como solução para o negócio, visto que todos os interessados "*Stackholders*" estão sistematicamente envolvidos.

Segundo [Batista 2004], é importante filtrar o conjunto de dados coletados, separando os elementos mais importantes, para que sejam transformados em informações realmente úteis. No seu ponto de vista a informação é o resultado do tratamento dos dados, sendo estes convertidos em um contexto significativo e útil para usuários finais específicos. Além da importância de filtrar os dados para a obtenção de informações relevantes, é ainda necessário que o sistema seja intuitivo, fácil de usar e que seus usuários, por meio do sistema, consigam realizar suas tarefas.

Em relação a usabilidade do sistema, [Paula Filho 2009] afirma que a falta de habilidade dos usuários e a dificuldade de operar um sistema podem causar diversos problemas. O autor esclarece que manter a usabilidade é fundamental para que o sistema realize o seu principal papel, transformar dados em informações úteis. [Pressman 2011] aponta que o sistema deve passar por testes de usabilidade, pois estes determinam o grau com o qual a interface do aplicativo facilita a vida do usuário. Neste sentido, [Guedes 2009] afirma que a interface com o usuário deve ser amigável, isto é, fácil de usar, simples e objetiva.

No desenvolvimento de um *software* quando o cliente não especifica todas as funções ou tarefas que o sistema deve realizar, a prototipagem é a melhor metodologia de desenvolvimento a ser adotada. [Pressman 2011] afirma que a prototipagem é a melhor opção quando os objetivos e funcionalidades do sistema não são definidos detalhadamente. Para [Rezende 2005], o ciclo de vida prototipagem reduz significativamente o tempo de desenvolvimento e a necessidade de manutenção, e ainda, explica que esse ciclo de vida é utilizado quando o comportamento do usuário é imprevisível.

De acordo com [Pressman 2011], os requisitos iniciais do projeto devem ser levantados por meio de reuniões entre o cliente e o engenheiro de software, neste contato inicial, deve ser levantado todas as informações mais importantes e funcionalidades mínimas obrigatórias. Logo após, um projeto básico deve ser implementado, sendo este, o primeiro protótipo. No entendimento de [Guedes 2009], as entrevistas com o cliente são cruciais para conhecer as necessidades reais do mesmo, de forma a revelar as funcionalidades que o sistema deve conter. Ainda neste contexto, [Bezerra 2007] afirma que o levantamento de requisitos tem como objetivo conhecer o problema do cliente e entender quais processos devem ser automatizados.

O levantamento de requisitos, de acordo com [Melo 2010] é difícil de ser realizado, visto que muitas vezes o cliente não sabe expressar-se de maneira clara e objetiva. [Guedes 2009] explica que todo sistema independente de sua complexidade, deve ser modelado antes de iniciar a codificação, isto se deve pelo dinamismo de um projeto, pois o cliente quase sempre solicita novas funcionalidades, o que torna o mesmo mais complexo, lento e dispendioso.

Após o levantamento de requisitos é necessário definir uma linguagem de modelagem de *software*. Para isto, existe como opção a *Unified Modeling Language* (UML), que trata-se de uma linguagem unificada de modelagem orientada a objetos utilizada como apoio às práticas de engenharia de software. Segundo [Rezende 2005], a linguagem UML possui diagramas com características próprias e com objetivos bem definidos. Logo, [Guedes 2009] afirma que o diagrama mais informal é o diagrama de caso de uso, sendo este, utilizado no momento em que são levantados os requisitos iniciais. [Bezerra 2007] aponta que o diagrama de caso de uso é o mais fácil de ser utilizado e também é o que fornece uma visão geral do sistema. [Lima 2011] comenta que o diagrama de caso de uso descreve o comportamento que o sistema tem em função da interação com os atores, mas sem revelar a estrutura interna do sistema.

Segundo [Guedes 2009], o diagrama da UML mais importante e mais utilizado é o diagrama de classes. É nele que são definidos os atributos e métodos que farão parte do sistema. Neste contexto, [Pressman 2011] explica que o diagrama de classes é o modelo que visa solucionar o problema a partir de uma visão estática da estrutura do sistema. Também afirma que o diagrama de classes descreve todos os objetos e relacionamentos estáticos envolvidos no sistema.

O diagrama de sequência, de acordo com [Guedes 2009], tem como objetivo descrever a comunicação entre os objetos por meio de mensagens, tendo como referência o diagrama de classes e caso de uso. [Pressman 2011] afirma que o diagrama de sequência mostra as interações em um caso de uso e os passos envolvidos na comunicação entre os objetos. Nesse sentido, [Rezende 2005] comenta que no diagrama de sequência os detalhes do funcionamento não são importantes e por isso não são descritos.

Após a elaboração dos diagramas de caso de uso, classe e sequência, faz-se necessário adotar alguma linguagem para estruturar as páginas do sistema *Web*. Para isto, existe a linguagem de marcação *Hipertext Markup Language* (HTML), que tem a função de representar graficamente elementos como textos e imagens em documentos veiculados na *internet*. Para [Oliviero 2007], a linguagem HTML funciona como uma espécie de sinal de trânsito, pelo qual uma página *Web* informa aos *browsers* (navegadores) como os elementos da página devem ser exibidos na tela do usuário.

Para [Silva 2008], a linguagem HTML serve para criar a estrutura das páginas *Web* utilizando hipertextos. Apesar desta linguagem de marcação ser muito importante e disponibilizar também de recursos de formatação, sozinha não consegue suprir as necessidades técnicas exigidas atualmente. Devido a este fato, surgiram linguagens com a função de criar estilos para as páginas de *internet*, utilizando recursos que a linguagem HTML não possui. Um exemplo é a linguagem de estilização denominada *Cascading Style Sheet* (CSS), que significa "folha de estilos em cascata".

[Silva 2012] afirma que a linguagem CSS possui recursos que HTML não possui, sendo assim, utilizada para enriquecer as páginas *Web*, fornecendo um visual mais bonito e muito mais dinâmico, tornando possível criar estilos para diversas páginas com um único arquivo separado das outras camadas do sistema, que com uma simples alteração no código de estilo, várias páginas podem ser modificadas instantaneamente e sem muito trabalho do desenvolvedor. De acordo com [Silva 2012], a linguagem CSS tem como objetivo devolver ao HTML seu propósito inicial, que era marcar e estruturar os conteúdos das páginas.

Além das linguagens de marcação e estilização, é possível fazer uso de uma ferramenta muito utilizada por desenvolvedores, o *framework Bootstrap*, que une funções das linguagens HTML, CSS e *JavaScript*. Esta ferramenta pode ser utilizada para garantir a responsividade do sistema *Web*. Para [Magno 2012], o *Bootstrap* é muito fácil de usar e facilita o desenvolvimento *front-end*, possui visual moderno e outra grande vantagem é que foi criado para desenvolvimento responsivo.

Outro recurso utilizado para o desenvolvimento *Web* é o *JavaScript*, que segundo [Morrison 2008] é uma linguagem de programação de *scripts* que tem como objetivo interagir com o lado do cliente (*browser*), é muito utilizado para validar informações inseridas em campos de texto e para realizar cálculos. Existem bibliotecas *JavaScript* que também facilitam o desenvolvimento, um exemplo é o j*Query*. Esta ferramenta, de acordo com [Silva 2009] fornece funcionalidades interativas por meio de *scripts* e é muito utilizada pelos desenvolvedores de sistemas *Web*. Com a utilização da biblioteca j*Query* pode-se atribuir eventos, definir efeitos, alterar ou criar elementos na página de maneira relativamente simples.

Em relação às linguagens de programação para desenvolvimento de sistemas *Web*, existe como opção o *Hipertext Preprocessor* (PHP), que de acordo com [Dall'Oglio 2007] é uma linguagem de programação para criação de aplicações *Web*, criada por Rasmus Lerdorf, sendo uma das mais utilizadas e populares do mundo. [Niederauer 2004] comenta que o PHP é popular por ser fácil de programar, eficiente e de código aberto. Porém, para [Dall'Oglio 2007], o PHP precisava de melhorias para a questão de OO (Orientação a Objetos). Desde então, a linguagem teve diversas melhorias neste quesito.

Para [Welling 2005], a OO faz uso de conceitos de classes e objetos, criando-se uma analogia com o mundo real. Para [Dall'Oglio 2007], é uma estrutura que visa descrever objetos por meio de seus atributos e métodos. [Dall'Oglio 2007] explica que os atributos são as características próprias desses objetos e os métodos seriam suas funções, isto é, o que ele pode realizar ou executar. Reforçando essa ideia, [Dall'Oglio 2007] aponta que a OO é uma maneira de aproximar a linguagem de programação a conceitos do mundo real, fazendo com que diversos objetos de classes diferentes troquem informações entre si.

Em muitos sistemas informativos é necessário manipular e armazenar dados, para isso existem os chamados Sistemas Gerenciadores de Banco de Dados (SGBD). Segundo [Oliviero 2011], SGBD's são sistemas que gerenciam banco de dados e tem como objetivo controlar o seu acesso, manipulação e organização. [Silberschatz 2006] afirma que é necessário fazer uso de um SGBD para gerenciar as informações com segurança. Neste ponto de vista, [Elmasri 2005] explica que o mais importante aspecto de um SGBD é ter a capacidade de manutenção e proteção das informações que ele gerencia.

Para [Oliviero 2011], o *MySQL* é uma excelente ferramenta para gerenciar banco de dados e possui diversos recursos compatíveis com outros grandes SGBD's. [Milani 2007] afirma que a ferramenta *MySQL* possui recursos que todos os seus concorrentes possuem, mas com a grande vantagem de ser gratuito. Além disso, o *MySQL* é mais fácil de programar e possui funções simples em relação a outros sistemas similares.

Uma ferramenta muito útil na modelagem do banco de dados *MySQL* é o *MySQL Workbench 6.1 CE,* devido aos seus recursos de engenharia reversa. No entendimento de [Feltrim 1999], a engenharia reversa é um procedimento técnico que analisa um objeto, dispositivo ou *software*, buscando entender o seu funcionamento e estrutura para fins de manutenção ou para criação de um novo produto com as mesmas características. O autor ainda afirma que na engenharia reversa de *software*, o foco é criar visões do sistema em diferentes níveis de abstração, de modo a facilitar o seu entendimento, com o objetivo principal de facilitar a manutenção do sistema.

Para acessar e manipular os dados em um sistema SGBD é necessário utilizar uma linguagem. Para isso, utiliza-se a linguagem *Structured Query Language* (SQL) que significa "Linguagem de Consulta Estruturada". De acordo com [Oliviero 2011], a linguagem SQL foi desenvolvida pela *International Business Machines* (IBM), na década de 70 para computadores de grande porte. É utilizada para consulta, atualização, criação e gerenciamento de bancos de dados relacionais. [Silberschatz 2006] comenta que a SQL se tornou a linguagem padrão de banco de dados relacionais. O instituto *American National Standards Institute* (ANSI) juntamente com a *International Organization for Standardization* (ISO) publicaram o primeiro padrão SQL, chamado (SQL-86) em 1986 pelo instituto ANSI e no ano seguinte pela ISO.

Além da escolha de um sistema gerenciador de banco de dados adequado ao projeto, é necessário adotar um padrão de arquitetura de software, onde um dos modelos mais utilizados é o *Model View Controller* (MVC). Para [Freeman 2007], é um padrão arquitetural que separa uma aplicação em três camadas diferentes. A *Model* é responsável pela regra do negócio, é nela que são contidas as funções do sistema e a comunicação com o banco de dados. Logo, a *View* é responsável pela visualização do sistema, representa a interface com o usuário e é por esta camada que o usuário interage com o sistema. A camada *Controller* é a ponte entre a *View* e *Model*, é responsável pelo fluxo das informações no sistema. Ainda neste contexto, [Dall'Oglio 2007] explica que uma aplicação organizada no padrão MVC oferece vantagens como a fácil manutenção e a reutilização de código. Outro ponto importante do padrão MVC é que a *View* fica independente das outras camadas, isto é, não interfere nas regras de negócio ou no fluxo de dados.

Além das escolhas de ferramentas de desenvolvimento e padrão de projeto, é necessário verificar a eficiência do *software* por meio de testes, que segundo [Pressman

2011], são necessários para encontrar possíveis erros e consequentemente buscar soluções. [Pressman 2011] ainda explica que o teste "Caixa Preta" é um teste comportamental que tem como foco verificar os requisitos funcionais do sistema, verificando as funções incorretas ou inexistentes, erros de interface, erros em estruturas de dados, erros de comportamento ou desempenho e erros de inicialização ou término.

3. Materiais e Métodos

Para o desenvolvimento deste projeto foi adotado a prototipagem como ciclo de vida. Logo após, foi levantado os requisitos do sistema por meio de entrevistas com um representante da Faculdade Guairacá – Guarapuava – PR. Por meio destas, foram definidas as funcionalidades e objetivos do projeto, observadas no Apêndice A.

A funcionalidade de maior relevância observada foi o tratamento dos dados informados pelo usuário e suas respectivas validações em relação à disponibilidade dos membros e a verificação da validade das datas e horários das bancas avaliadoras. Este requisito tem prioridade devido à dificuldade de gerenciar essas informações com editores de texto ou com planilhas eletrônicas. A Faculdade Guairacá não possui nenhum aplicativo que gerencie as bancas avaliadoras de trabalhos de conclusão de curso. O controle é realizado por meio de uma planilha eletrônica, que pode ser observada no Anexo A.

Após a coleta dos requisitos e a definição do ciclo de vida do projeto, foram desenvolvidos três diagramas UML, utilizando-se do aplicativo *Astah Community 6.9.0*. Foi criado o diagrama de caso de uso e apresentado ao cliente as possíveis interações do usuário com o sistema, bem como suas funcionalidades. O diagrama pode ser observado no Apêndice B.

Em seguida foi elaborado o diagrama de classes, que pode ser visto no Apêndice C. Após a sua análise, juntamente com o cliente foi decidido que todos os membros das bancas deveriam ser inicialmente usuários comuns e que seus papéis seriam determinados no momento em que uma banca fosse cadastrada. Para solucionar esse problema, foi adicionado a classe *BancaUsuario* que seria a responsável por atribuir o papel de cada membro participante da banca.

Ainda no contexto de UML, foi desenvolvido o diagrama de sequência, este teve o papel de apresentar ao cliente as funções do sistema por meio de uma sequência de passos, demonstrando como uma banca e seus membros são cadastrados e validados. O diagrama de sequência pode ser observado no Apêndice D.

Paralelamente aos diagramas UML, foi necessário modelar o banco de dados, que promoveu toda a persistência dos dados no sistema. Para isto, foi utilizado o aplicativo *MySQL Workbench 6.1 CE*. Esta ferramenta teve papel fundamental no projeto, pois por meio desta foi possível determinar os relacionamentos entre as tabelas que representam as classes e seus atributos. Esta ferramenta possui recursos que facilitaram o desenvolvimento do sistema, um deles foi a possibilidade de realizar engenharia reversa. Isto é, poder transformar um modelo diagramático em um *script* SQL e vice-versa. A modelagem do banco de dados pode ser vista no Apêndice E.

O primeiro passo para a codificação foi a definição dos recursos de visualização das páginas HTML. Para isto, foi utilizado o *framework bootstrap* e feito uso da linguagem CSS para realizar as mudanças de *layout* no decorrer do projeto. Em princípio, a responsividade não estava contida nos requisitos iniciais, mas foi sugerido

ao cliente devido às vantagens de se poder acessar o sistema de dispositivos móveis como *smartphones*, *tablets*, entre outros. O *bootstrap* teve papel fundamental para tornar o sistema responsivo.

O sistema foi dividido em camadas, conforme o padrão de projeto MVC. As camadas foram representadas por pastas com o nome que as representa. Sendo elas, v*iew, model e controller*. Os arquivos com formato HTML foram inseridos na pasta "*view"*. Logo, as classes foram armazenadas na pasta "*controller"* e as classes que criam a conexão com o banco de dados foram inseridas na pasta "*model"*. Outras pastas foram criadas na raiz do projeto para separar os arquivos por funcionalidades. As imagens, arquivos de estilos e *scripts* foram armazenados na pasta "*public"*.

Neste projeto, fez-se uso do paradigma de programação orientado a objetos e o PHP como linguagem de programação. A versão 5 do PHP possui muitos recursos para desenvolvimento OO. A grande vantagem deste paradigma é a facilidade de manutenção do sistema, pois é possível reutilizar o código. Além disso, com este conceito foi possível aproximar a programação da vida real e o ciclo de vida do sistema torna-se mais longo.

Para atender os requisitos do sistema em relação aos relatórios e certificados, fez-se uso da biblioteca mPDF. Esta ferramenta foi de grande valia pela facilidade de implementação e pela possibilidade de utilizar estilos da linguagem CSS para formatar os relatórios, dessa forma foi possível mudar as cores e fontes dos textos, inserir paginação, cabeçalho e rodapé.

No decorrer do projeto, ocorreram mudanças em relação aos requisitos iniciais coletados. Em algumas apresentações de protótipos, observou-se a necessidade de incluir algumas funções que não foram definidas no primeiro levantamento de requisitos. O primeiro protótipo pode ser observado no Apêndice F. Uma mudança importante nos requisitos foi o controle de usuários, que tem como objetivo gerenciar as permissões de acesso ao sistema de acordo com o perfil de cada usuário. Em princípio, o sistema seria usado apenas pelo departamento de Análise de Sistemas da Faculdade Guairacá, porém decidiu-se que seria melhor ampliar a utilização para os demais departamentos. Inicialmente, o sistema controlará quinze cursos de graduação da instituição, onde cada gestor tem permissão de acesso somente ao seu curso, sem interferir no trabalho dos demais departamentos. É neste ponto que o controle de usuários tem papel fundamental.

Em reuniões com o cliente, cogitou-se a possibilidade de permitir que visitantes do sistema pudessem cadastrar-se como ouvinte em qualquer banca. Nos requisitos iniciais, a função de cadastrar ouvintes nas bancas cabia ao gestor do curso. Esta nova funcionalidade foi implementada, permitindo que visitantes se cadastrem como ouvinte em qualquer banca por meio do sistema. Este recurso pode ser observado no Apêndice G. Outra mudança foi a inclusão de cadastros de Gestores. O cadastro de Gestores pode ser observado no Apêndice H. Definiu-se que cada curso deveria possuir um gestor, que seria o responsável pelos cadastros e manutenção das bancas do curso em questão. Observando que o mesmo não tem permissão de realizar qualquer modificação ou cadastro de bancas pertencentes a outros cursos.

Definiu-se também que o único usuário com acesso irrestrito ao sistema é aquele com perfil de administrador. Isto posto, foi implementado o cadastro de administradores e uma função que permite a retirada de seus privilégios, caso necessário. Este novo recurso pode ser observado no Apêndice I. Outra nova funcionalidade adicionada foi a

possibilidade de alteração de senha de usuário mediante informação da atual, o que pode ser visto no Apêndice J.

Em relação às consultas das bancas, foram criados filtros com diversas opções de filtragem. Esta necessidade foi observada devido à ampliação do uso do sistema a diversos departamentos da Faculdade Guairacá. Essa modificação foi realizada no módulo de gestão e administração. A tela de pesquisa pode ser observada no Apêndice K. Um exemplo de declaração de participação e lista de presença pode ser visto no Apêndice L e M, respectivamente.

Após a implementação do sistema foi utilizado o teste "Caixa Preta" para identificar possíveis erros ou comportamentos inesperados. Em seguida, foi elaborado uma tabela de avaliação que foi preenchida pelo cliente, podendo ser observada no Apêndice N.

4. Resultados e Discussões

Com o Sistema Gerenciador de Bancas foi possível controlar e manter as bancas avaliadoras dos trabalhos de conclusão de curso da Faculdade Guairacá de forma ágil e eficiente, diminuindo radicalmente as chances de erros nos cadastros e nas suas atualizações. Com o sistema, é possível consultar qualquer banca por meio de filtros bem definidos e realizar ações como inclusão, exclusão, edição, emissão de relatórios, certificados e controle de usuários. Antes da implementação do sistema, o processo de emissão de relatórios e certificados eram realizados fazendo uso de editores de texto ou planilhas, um a um, o que demandava tempo e trabalho. Com a implementação do sistema, esse processo tornou-se automatizado, deixando o processo mais rápido, fácil e seguro.

Com o uso do sistema, os gestores dos cursos não precisam mais verificar a disponibilidade dos membros das bancas e nem mesmo checar os dias e horários disponíveis para uso das salas que normalmente são usadas para as apresentações das bancas. Antes da implementação do sistema, este processo era realizado por meio de contatos pessoais, o que consumia tempo e trabalho. Agora, no momento do cadastro, se um membro ou sala não estiver disponível, uma mensagem na tela surgirá informando essa indisponibilidade.

Em muitas bancas avaliadoras, ouvintes que chegavam atrasados para a apresentação eram impedidos de participar por não ter mais lugares disponíveis ou pelo fato da banca já ter sido iniciada. Com a implementação do sistema, somente os ouvintes que se cadastraram poderão assistir uma banca. Uma lista de presença é gerada automaticamente, contendo os nomes dos participantes da banca em questão. Sendo assim, o ouvinte fica ciente das regras de participação, como data, horário e sala. O sistema impede que ouvintes cadastrem-se em bancas que excederam o limite de participantes. Este limite de ouvintes é determinado no momento do cadastro da banca, baseado no tamanho da sala em que a apresentação ocorrerá. Com essas medidas, são evitados problemas como os já mencionados.

Antes da implementação do sistema, os coordenadores dos cursos de graduação precisavam estar em seus postos de trabalho para terem acesso às planilhas eletrônicas, documentos de texto ou outros controles obsoletos a fim de organizar e agendar as bancas avaliadoras. Agora, com o Sistema Gerenciador de Bancas, isso pode ser feito de qualquer lugar onde é possível o acesso à internet. Além disso, o gestor não precisa estar em frente ao seu computador de trabalho, agora ele pode realizar todas essas tarefas no smartphone ou tablet em qualquer lugar e horário.

Fazendo-se uso dos conceitos de engenharia de software e da UML foi possível perceber as funcionalidades adicionais que o sistema necessitava conter. Os diagramas UML foram fundamentais para chegar a um entendimento sobre os requisitos e soluções de problemas juntamente com o cliente. Por meio destas ferramentas que foi possível solucionar todos os problemas encontrados. A escolha do ciclo de vida também foi essencial, já que tornou possível apresentar ao cliente modelos reais e funcionais por meio de protótipos. Assim, tornou-se possível criar um diálogo produtivo com o cliente, de maneira a chegar a um entendimento sobre os propósitos, operações e funcionalidades do sistema.

4. Considerações finais

A realização deste projeto foi motivada pela dificuldade em controlar e manter as bancas avaliadoras dos trabalhos de conclusão de curso da Faculdade Guairacá. Em princípio, foi levantada a possibilidade de construção de um sistema *Web* apenas para cadastrar as bancas do curso de Tecnologia em Análise e Desenvolvimento de Sistemas. Após reflexão sobre o tema junto ao cliente, foi cogitado a ideia de incrementar mais funcionalidades ao sistema. Uma delas foi ampliar o uso do mesmo aos demais departamentos da instituição. Outras funcionalidades adicionais como cadastro de ouvintes, controle de usuários, alteração de senha, cadastro de gestores e administradores também foram implementadas ao longo do projeto.

O sistema foi desenvolvido fazendo uso da linguagem de programação PHP e do Sistema Gerenciador de Banco de Dados *MySql*. O aplicativo *MySQL Workbench 6.1 CE* foi de grande valia no decorrer do projeto por proporcionar uma interface amigável, o que facilitou a construção do banco de dados. Outras ferramentas como o *framework bootstrap*, linguagem CSS e a biblioteca j*Query* tiveram papéis importantes no desenvolvimento do sistema. Com a utilização do *bootstrap* foi possível manter a responsividade do sistema e uma interface gráfica agradável. Logo, a biblioteca j*Query* teve papel importante na validação dos campos dos formulários. Já a linguagem CSS foi fundamental para realizar mudanças no layout do sistema, tais como cores, fontes, texturas e contornos de elementos das páginas.

Conforme os usuários vão fazendo uso do sistema, surgem novas necessidades de recursos ou funcionalidades, o que é natural em se tratando de desenvolvimento de *softwares*. Sendo assim, no futuro serão realizadas atualizações do sistema para suprir essas possíveis necessidades que surgirão ao longo do tempo.

Referências

Batista, E. O. (2004). Sistemas de Informação: o uso consciente da tecnologia para o gerenciamento. Editora Saraiva.

Bezerra, E. (2007). Princípios de análise e projeto de sistemas com UML. Elsevier Editora.

Dall'Oglio, P. (2007). PHP Programando com Orientação a Objetos. Novatec Editora Ltda.

Elmasri, R. e Navathe, S. (2005). Sistemas de Bancos de Dados. Pearson Addison Wesley.

Feltrim, V. D. (1999). “Apoio à Documentação de Engenharia de Software por meio de Hipertexto “. Disponível em: http://www5.usp.br/?s=engenharia+reversa, Acessado em 10 de Julho de 2015.

Freeman, E. (2007). Use a Cabeça – Padrões de Projetos Design Patterns, volume 2ª Edição. Alta Books Ltda.

Guedes, G. T. A. (2009). UML 2: Uma abordagem prática, volume 3ª Edição. Novate Editora Ltda.

Lima, A. S. (2011). UML 2.3: Do requisito à Solução. Editora Érica Ltda.

Magno, A. (2012). “Globo bootstrap”. Disponível em: http://alexanmtz.github.io/bootstrap/index.html, Acessado em 4 de Junho de 2015.

Melo, A. C. (2010). Desenvolvimento de aplicações com UML 2.2: do conceitual à implementação. Brasfort Livros e Multimídia Ltda.

Milani, A. (2007). MySQL Guia do Programador. Novatec Editora Ltda.

Morrison, M. (2008). Use a Cabeça - Javascript. Alta Books Ltda.

Niederauer, J. (2004). Desenvolvendo web Sites com PHP: aprenda a criar websites dinâmico e interativos com PHP e banco de dados. Novatec Editora Ltda.

Oliveira, A. C. (2006). Inteligência Competitiva na Internet, volume 2ª Edição. Brasport Livros e Multimídia Ltda.

Oliviero, C. A. J. (2007). Faça um Site Orientado Por Projeto: HTML 4.0 Conceitos e Aplicações, Para webmasters e webdesigners. Editora Érica Ltda.

Oliviero, C. A. J. (2011). Faça um Site Orientado Por Projeto: PHP 5.2 com MySQL 5.0 Comércio Eletrônico. Editora Érica Ltda.

Paula Filho, W. d. P. (2009). Engenharia de Software: fundamentos, métodos e padrões. LTC.

Pressman, R. S. (2011). Engenharia de Software: Uma Abordagem Profissional, volume 7ª Edição. AMGH Editora Ltda.

Rezende, D. A. (2005). Engenharia de Software e Sistemas de Informação. Brasfort Livros e Multimidia Ltda.

Silberschatz, A. (2006). Sistema de Banco de Dados. Elsiever Editora Ltda.

Silva, M. S. (2008). Criando sites com html, sites de alta qualidade com html e css. Novatec Editora Ltda.

Silva, M. S. (2009). Ajax com jQuery Requisições AJAX com a simplicidade de jQuery. Novatec Editora Ltda.

Silva, M. S. (2012). CSS3: Desenvolva aplicações web profissionais com uso dos poderosos recursos de estilização das CSS3.Novatec Editora Ltda.

Welling, L. (2005). PHP e MySQL Desenvolvimento web. Elsiever Editora Ltda.

Apêndices

Apêndice A: Análise de Requisitos.

Análise de Requisitos para o Sistema de Gerenciamento de Bancas

Faculdade Guairacá.

Tela de Login: Deve conter uma tela de login e senha, para acesso do usuário ao sistema.

Cadastro de membros: Deve conter o nome, e-mail, celular, cpf, login, senha e tipo. O usuário somente terá acesso as bancas se estiver logado. Os membros podem ser autores, avaliadores, orientadores ou coorientadores. Deve permitir a inclusão, exclusão e alteração dos membros.

Cadastro de cursos: O formulário de cadastro deve conter o nome do curso e o departamento a que pertence. Deve permitir a inclusão, exclusão e alteração dos cursos.

Cadastro de ouvintes: Deve conter o nome, e-mail, celular, cpf, login, senha e tipo. Deve permitir a inclusão de ouvintes.

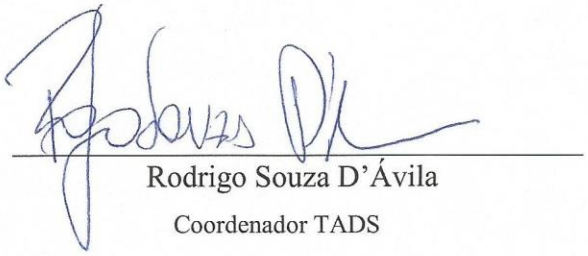
Certificados de participação de ouvintes: O sistema deve gerar certificados de participação de usuários ouvintes.

Declaração de participação de membros: O sistema deve gerar declarações de participação de membros da banca.

Lista de Presença: O sistema deve gerar uma lista de presença dos participantes das bancas. Deve conter o nome, cpf e campo para assinatura.

O sistema deve gerar uma lista geral de bancas. Contendo data, horário, título, sala, autor e orientador.

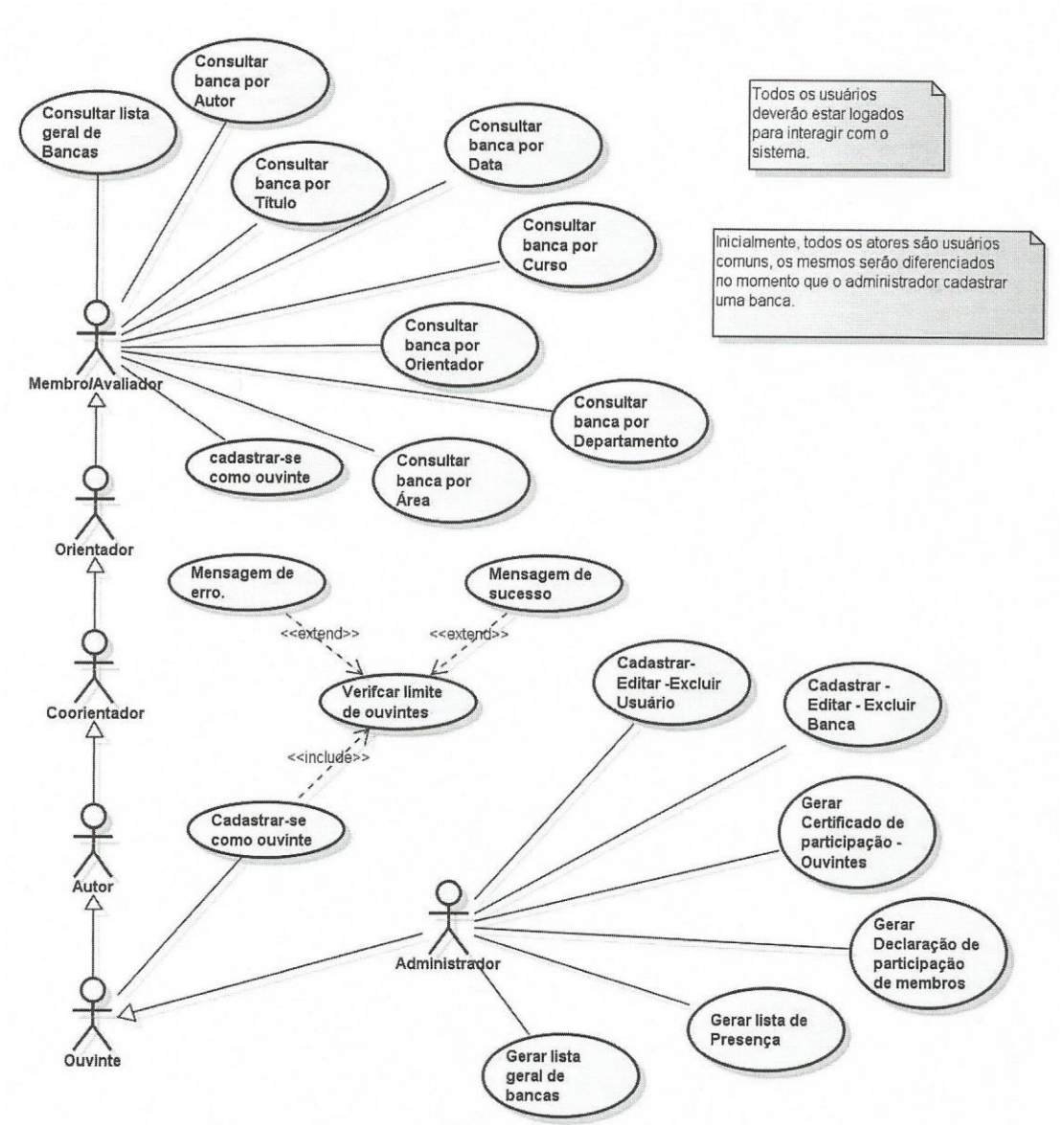
O sistema deve possuir uma interface simples e intuitiva, deve ser de fácil utilização.



Rodrigo Souza D'Ávila

Coordenador TADS

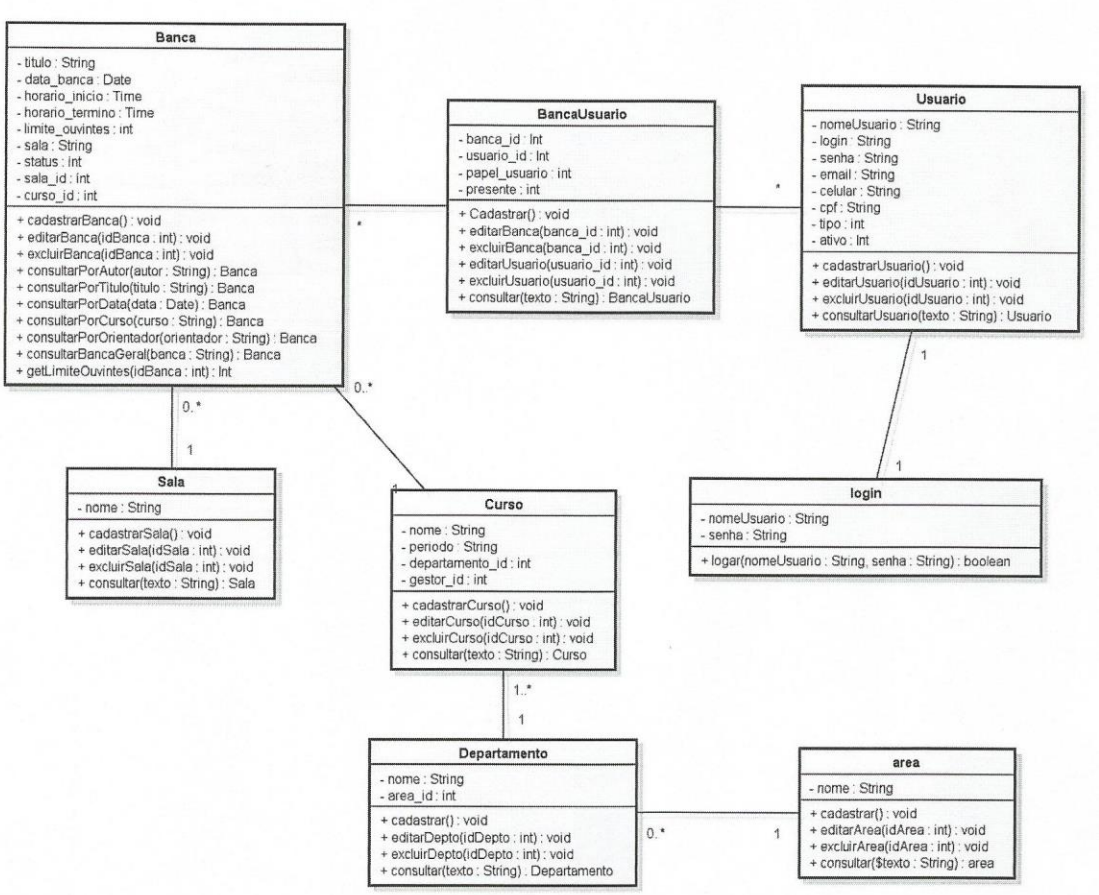
Apêndice B: Diagrama de caso de uso.



Rodrigo Souza D'Ávila

Coordenador TADS

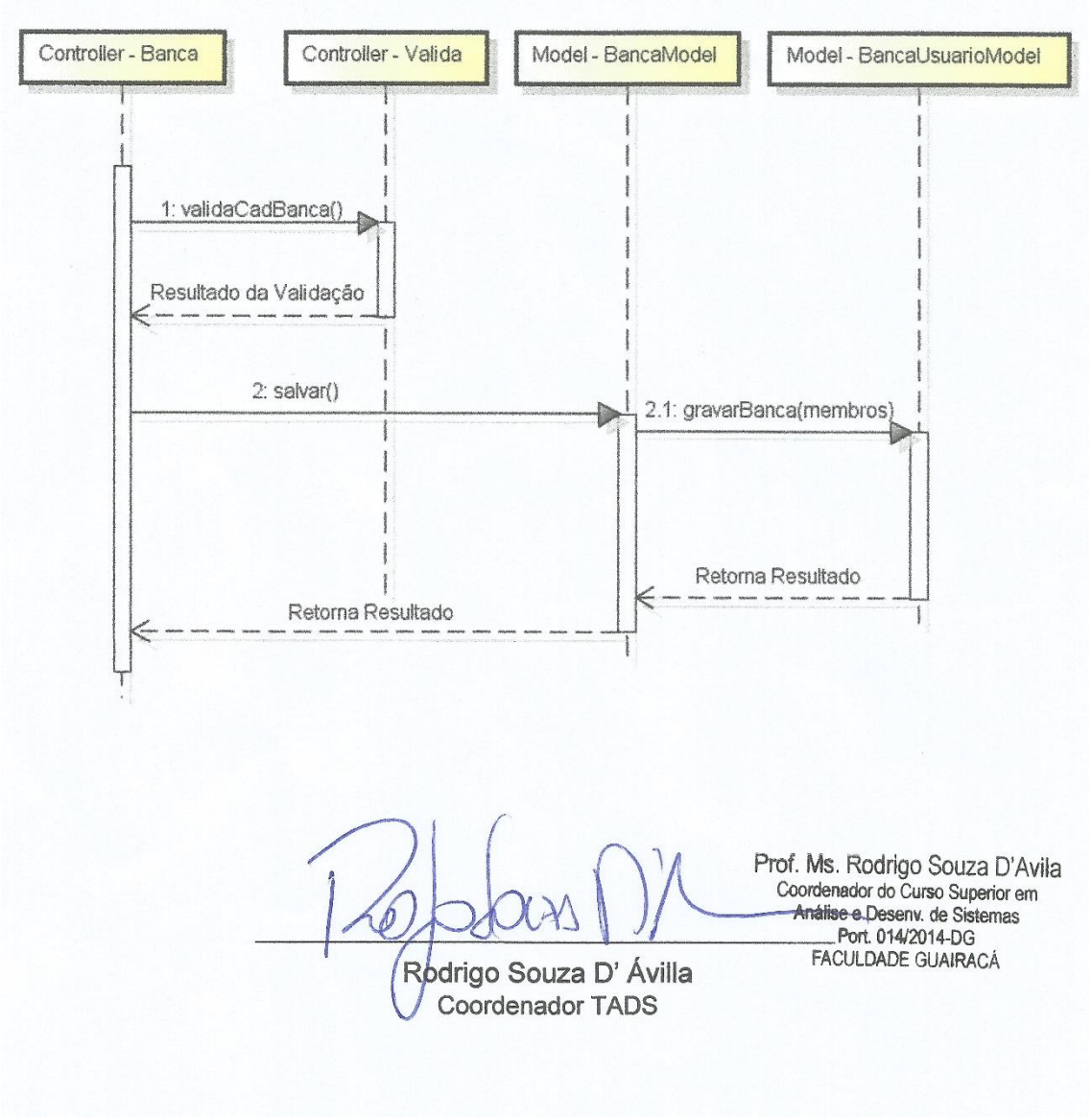
Apêndice C: Diagrama de classes.



Rodrigo Souza D'Ávila

Coordenador TADS

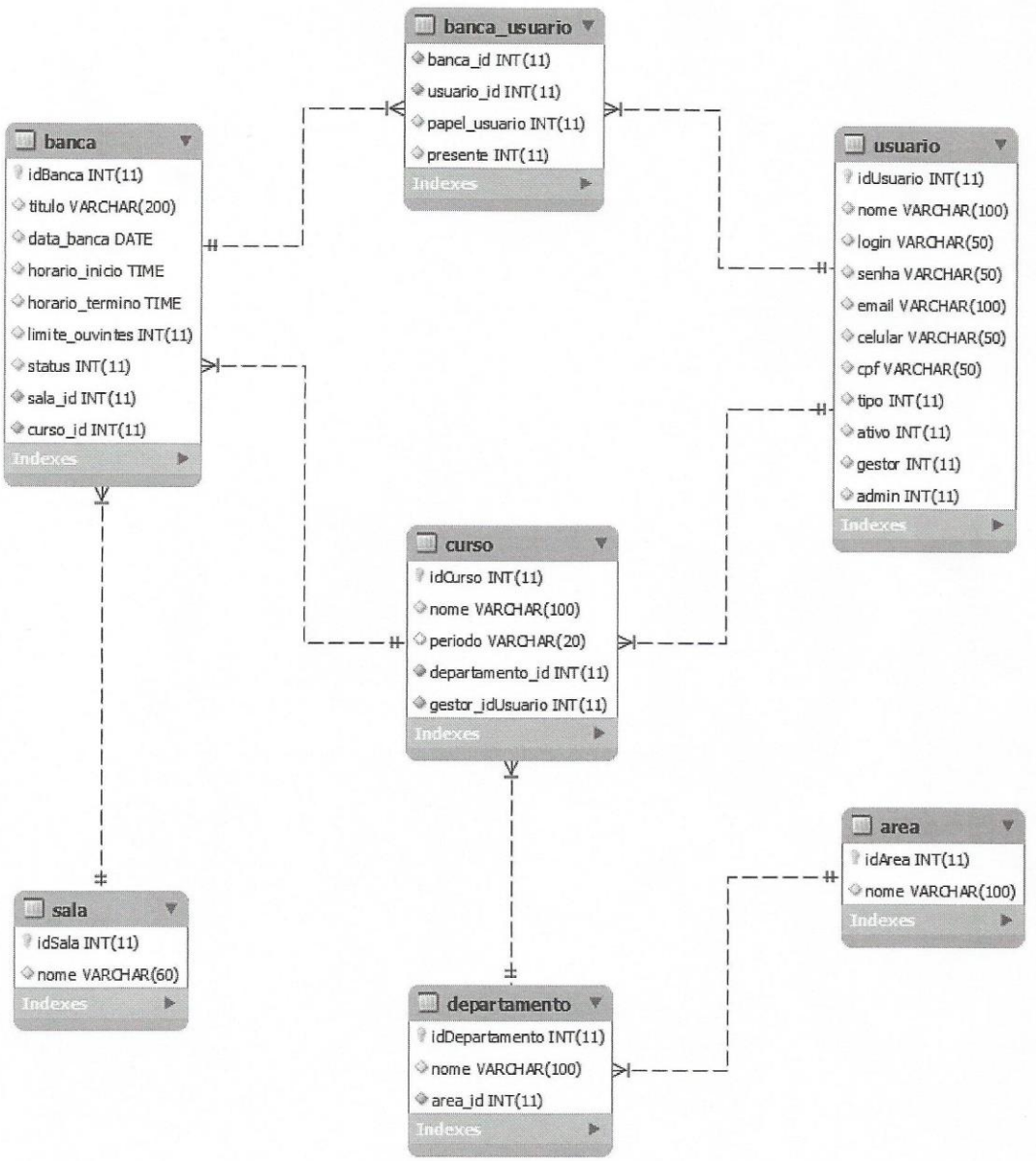
Apêndice D: Diagrama de sequência.



Prof. Ms. Rodrigo Souza D'Avila
Coordenador do Curso Superior em
Análise e Desenv. de Sistemas
Port. 014/2014-DG
FACULDADE GUAIRACÁ

Rodrigo Souza D' Ávilla
Coordenador TADS

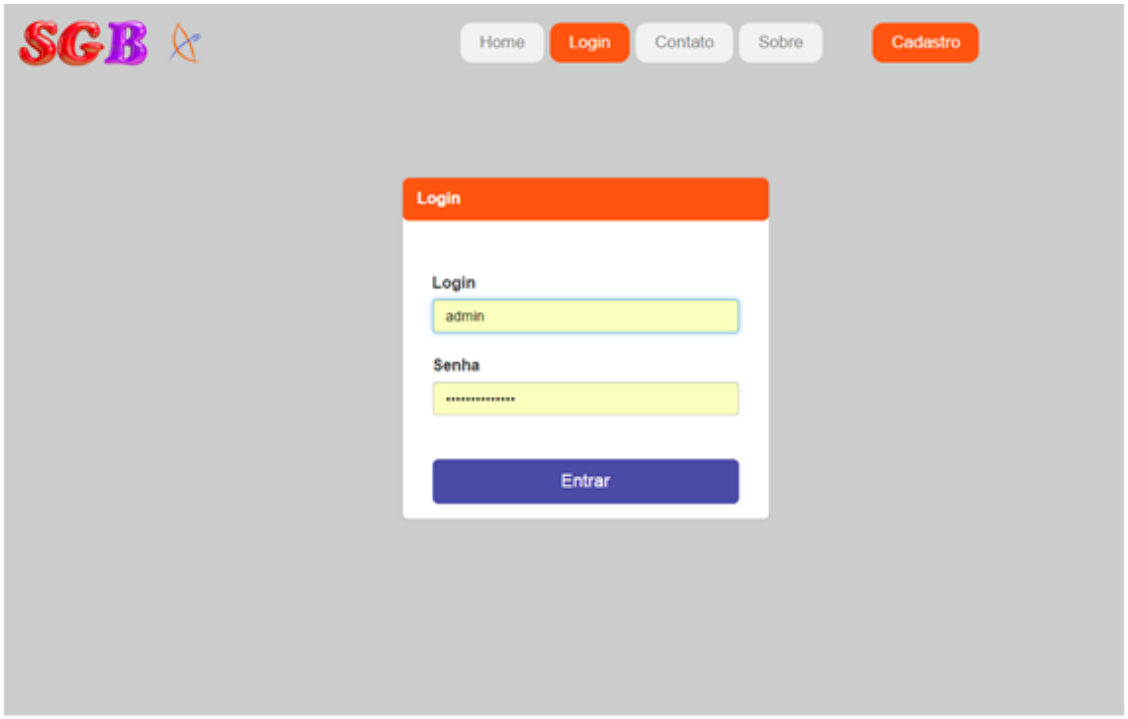
Apêndice E: Modelagem do banco de dados.



Rodrigo Souza D'Ávila

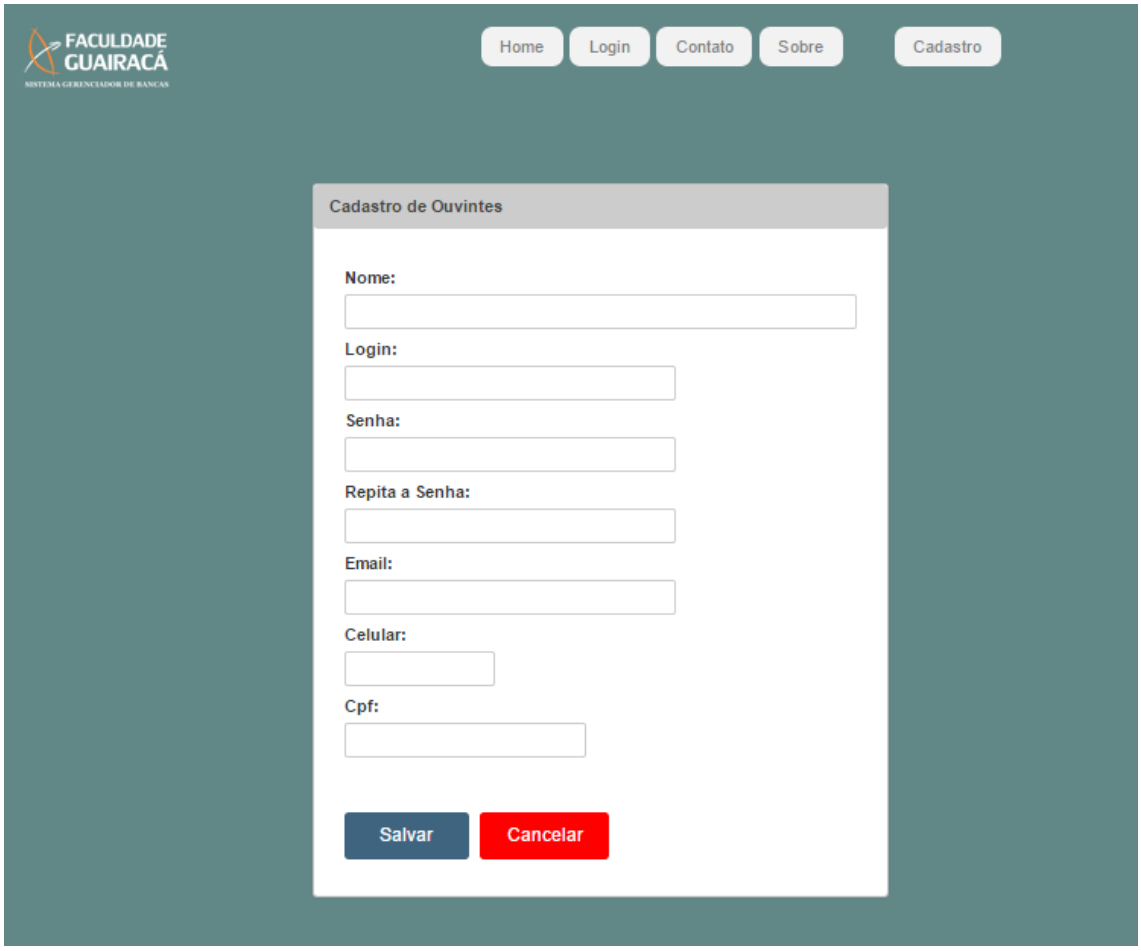
Coordenador TADS

Apêndice F: Primeiro protótipo.

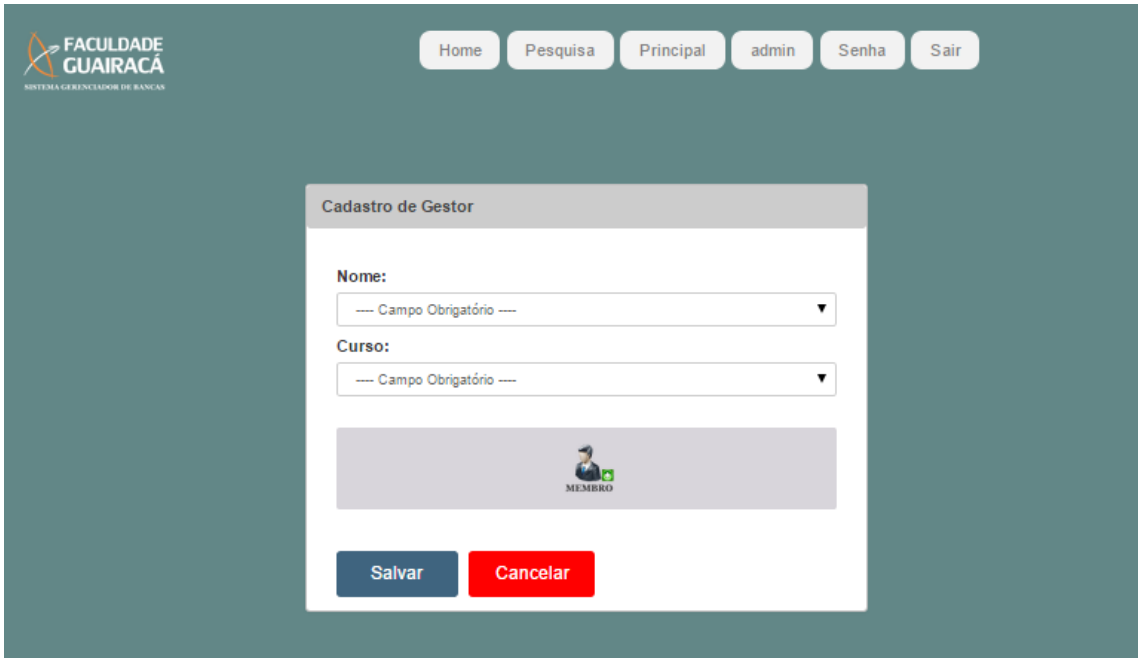


Rodrigo Souza D'Ávila
Coordenador TADS

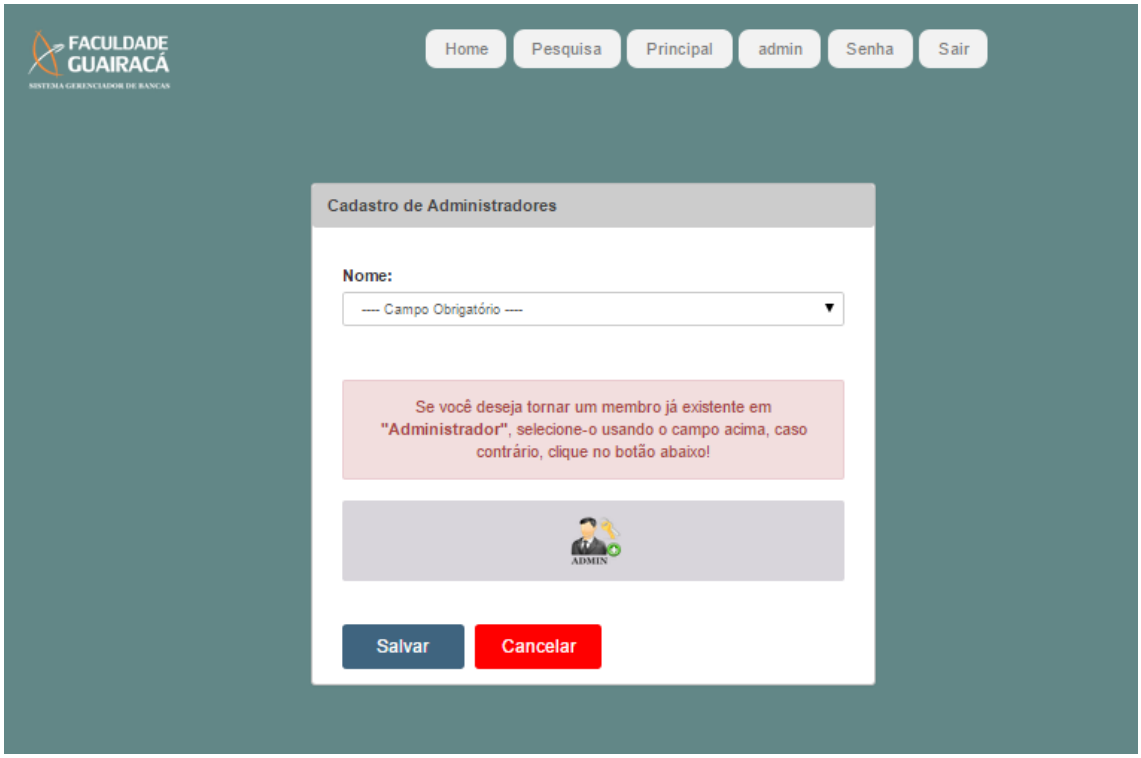
Apêndice G: Cadastro de ouvintes.

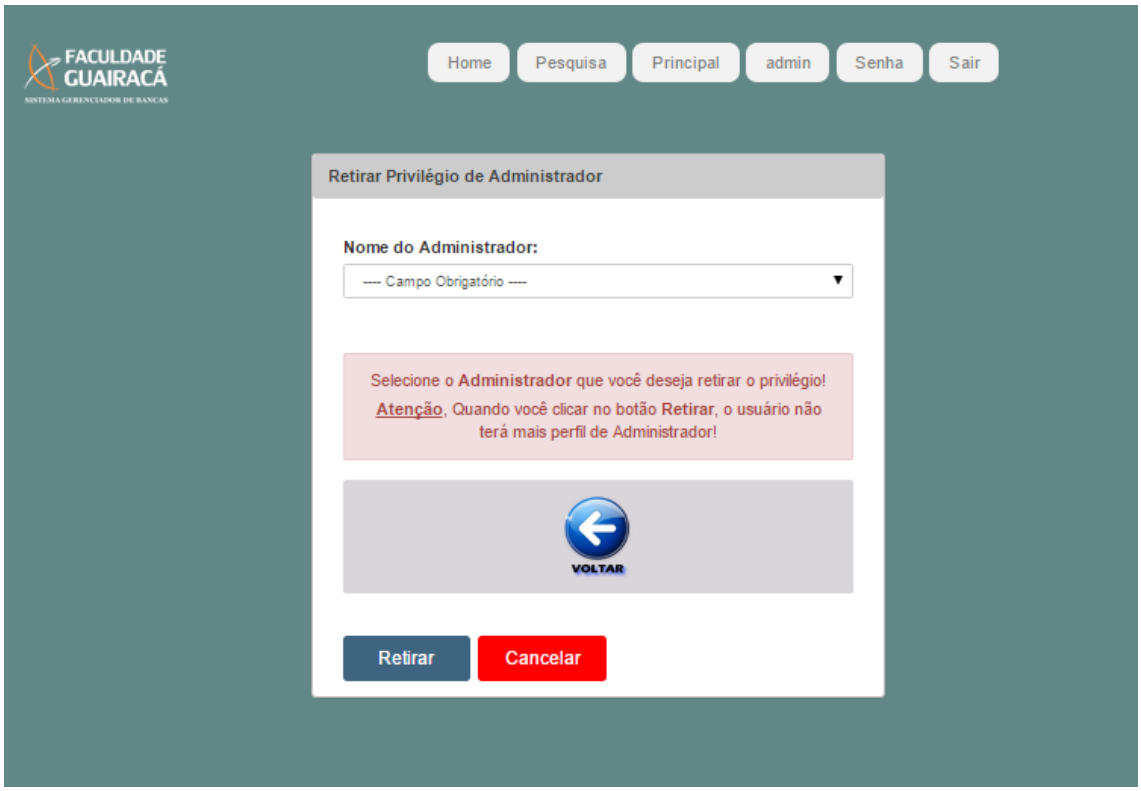


Apêndice H: Cadastro de gestores.

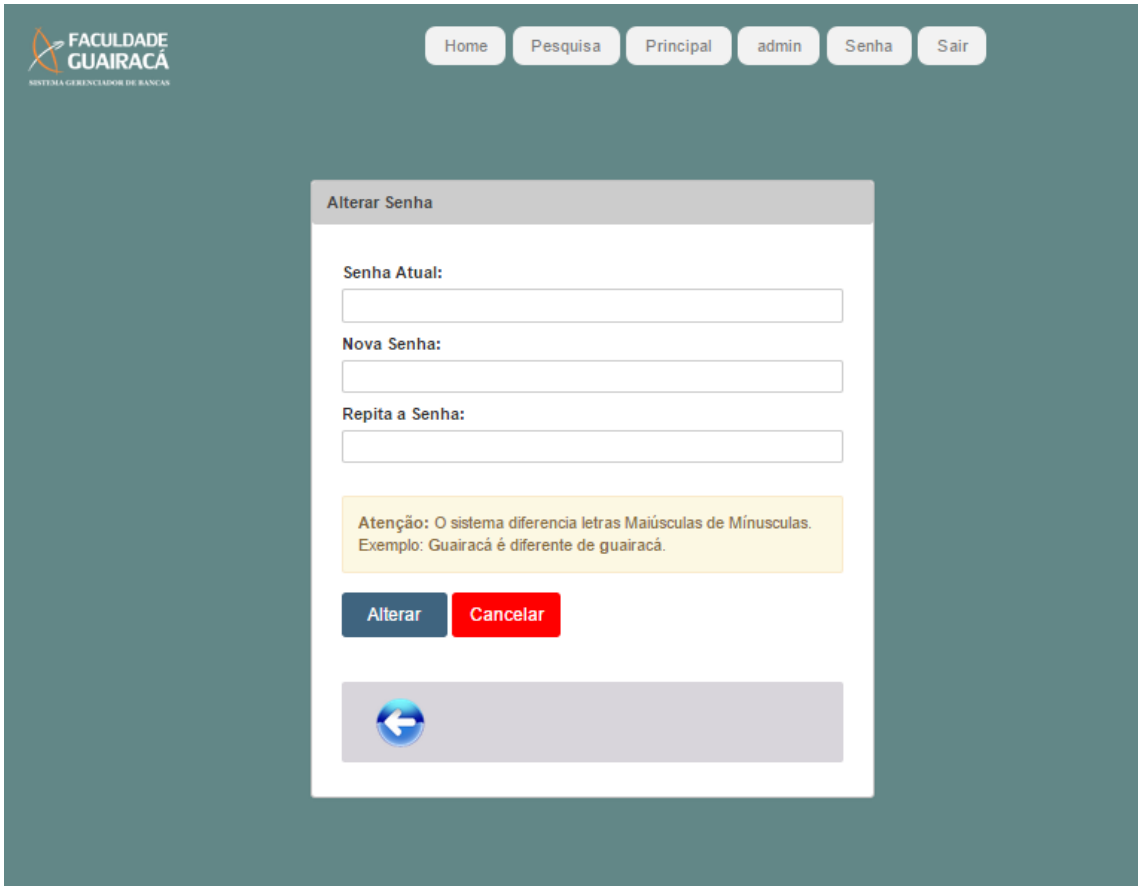


Apêndice I: Cadastro de administradores e retirada de privilégio.

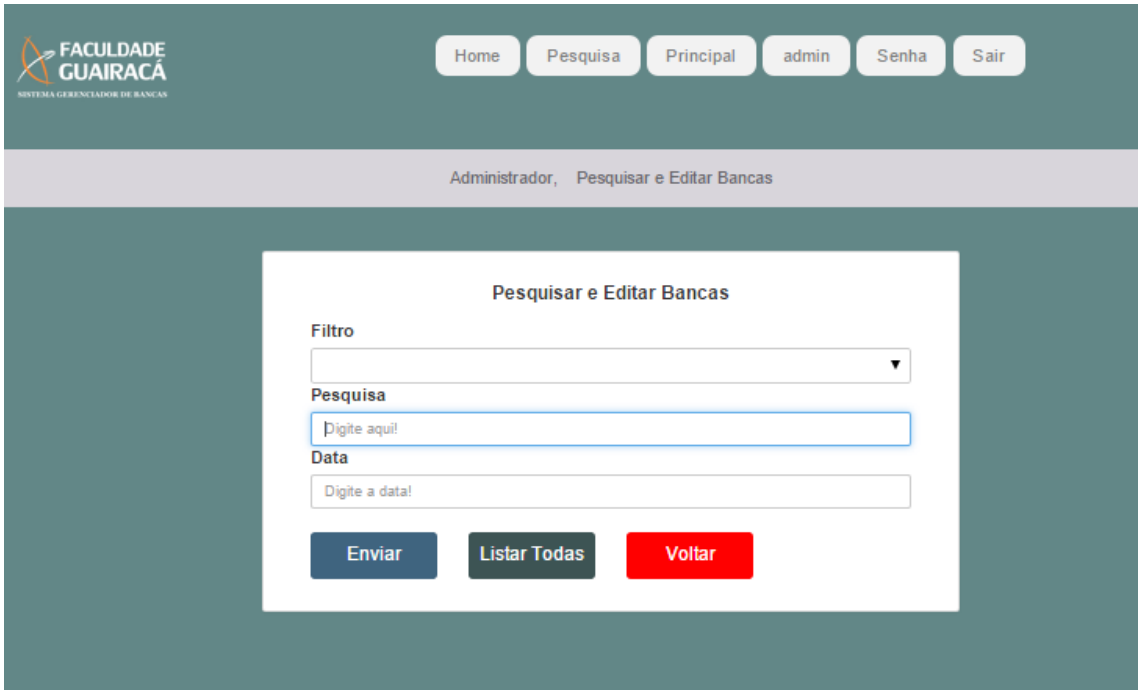


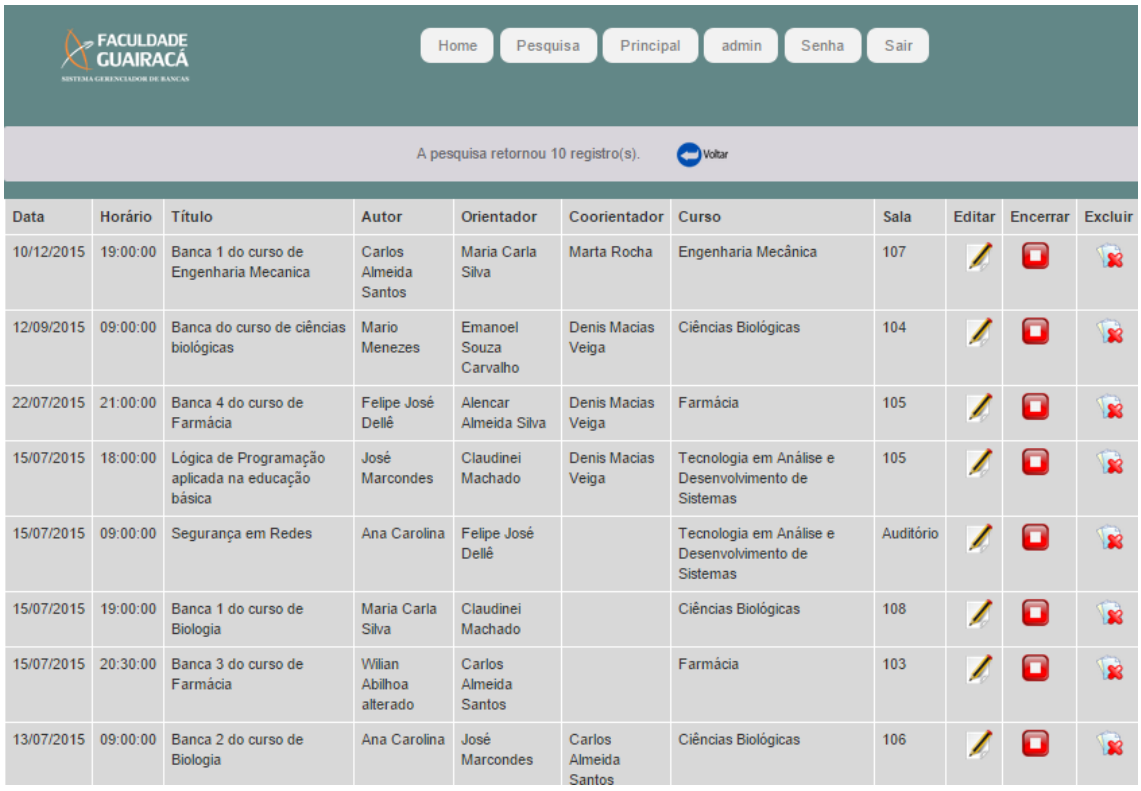


Apêndice J: Alterar senha.



Apêndice K: Pesquisa de bancas.





Data	Horário	Título	Autor	Orientador	Coorientador	Curso	Sala	Editar	Encerrar	Excluir
10/12/2015	19:00:00	Banca 1 do curso de Engenharia Mecanica	Carlos Almeida Santos	Maria Carla Silva	Marta Rocha	Engenharia Mecânica	107			
12/09/2015	09:00:00	Banca do curso de ciências biológicas	Mario Menezes	Emanoel Souza Carvalho	Denis Macias Veiga	Ciências Biológicas	104			
22/07/2015	21:00:00	Banca 4 do curso de Farmácia	Felipe José Dellê	Alencar Almeida Silva	Denis Macias Veiga	Farmácia	105			
15/07/2015	18:00:00	Lógica de Programação aplicada na educação básica	José Marcondes	Claudinei Machado	Denis Macias Veiga	Tecnologia em Análise e Desenvolvimento de Sistemas	105			
15/07/2015	09:00:00	Segurança em Redes	Ana Carolina	Felipe José Dellê		Tecnologia em Análise e Desenvolvimento de Sistemas	Auditório			
15/07/2015	19:00:00	Banca 1 do curso de Biologia	Maria Carla Silva	Claudinei Machado		Ciências Biológicas	108			
15/07/2015	20:30:00	Banca 3 do curso de Farmácia	Wilian Abilhoa alterado	Carlos Almeida Santos		Farmácia	103			
13/07/2015	09:00:00	Banca 2 do curso de Biologia	Ana Carolina	José Marcondes	Carlos Almeida Santos	Ciências Biológicas	106			

Apêndice L: Exemplo de declaração de participação.

Declaração

Guarapuava, 06 de junho de 2015

Declaro para os devidos fins que o professor **Rodrigo D'Avila**, portador do CPF **122.456.876-07** , participou como **Avaliador** da defesa do trabalho de conclusão do curso de "*Tecnologia em Análise e Desenvolvimento de Sistemas*" da Faculdade Guairacá, intitulado **Sistema de Gerenciamento de Bancas - Faculdade Guairacá** sob autoria do discente **Denis Macias Veiga**, defendido em **25/06/2015**.

_____

Regiane Orlovski

Registro	Folha	Livro

Apêndice M: Exemplo de lista de presença.

FACULDADE GUAIRACÁ
Seu momento é agora.

Gerado em: 16/06/2015

Lista de Presença

Titulo do Trabalho: **Sistema de Gerenciamento de Bancas - Faculdade Guairacá**
Data da Banca: **25/06/2015**
Aluno(Autor): **Denis Macias Veiga**
Orientador: **Felipe José Dellê**
Avaliador: **Rodrigo D'Avila**
Avaliador: **Regiane Orlovski**

Nome	Cpf	Assinatura
Alencar Almeida Silva	248.238.423-94	
Ana Carolina	727.188.888-24	
Denis Macias Veiga	118.452.338-08	
Eduardo de Oliveira	482.482.384-23	
Felipe José Dellê	827.778.234-24	
João Carlos José	772.882.348-82	
José Marcondes	924.923.942-39	
Marinaldo Antunes	131.231.232-13	
Regiane Orlovski	222.222.222-22	
Rodrigo D'Avila	122.456.876-07	

Apêndice N: Ficha de Avaliação.

Ficha de Avaliação

1	Utilização do Sistema	[x] Fácil [ ] Difícil
2	Ocorreu algum erro durante a execução das atividades?	[ ] Sim [x] Não
3	O sistema possui as funções de controle conforme o esperado?	[x] Sim [ ] Não
4	O sistema atende aos requisitos definidos no início do projeto?	[x] Sim [ ] Não
5	Como você classifica o tempo de resposta do sistema?	[x] Rápido [ ]Médio [ ] Lento
6	Em relação ao cadastro das bancas, o sistema realiza as validações necessárias?	[x] Sim [ ] Não
7	O sistema é eficiente em relação aos cadastros, edições, exclusões e consultas?	[x] Sim [ ] Não
8	Os relatórios e certificados estão sendo emitidos conforme o esperado?	[x] Sim [ ] Não
9	As telas do sistema estão bem distribuídas e possuem uma interface fácil de usar?	[x] Sim [ ] Não
10	O sistema é compatível com os navegadores mais utilizados?	[x] Sim [ ] Não
11	O menu do sistema facilita a navegação?	[x] Sim [ ] Não
12	O Sistema informa sobre erros de preenchimento de formulários e avisa quando ocorre algum erro ou exceção?	[x] Sim [ ] Não
13	Qual é o nível de melhoria que o sistema proporcionou em relação ao antigo controle?	[x] Alto [ ]Médio [ ] Baixo

Rodrigo Souza D' Ávila
Coordenador TADS

Prof. Ms. Rodrigo Souza D'Avila
Coordenador do Curso Superior em Análise e Desenv. de Sistemas
Port. 014/2014-DG
FACULDADE GUAIRACÁ

Anexos

Anexo A: Planilha utilizada para controle das bancas.

	ORIENTADO	ORIENTADOR/COORIENTADOR	PROFESSOR 1	PROFESSOR 2
1	CAMILA	ELENA	Sérgio	Raphael
2	CLAUDINEI	ELENA	Sérgio	Alexandre
3	ELOIR	REGIANE	Sérgio	Alexandre
4	BENTO	REGIANE	Rodrigo	Raphael
5	FELIPE SOCZEK	REGIANE	Elena	Alexandre
6	GUSTAVO K.	RODRIGO	Regiane	Wilian
7	SALLES	RODRIGO	Elena	Wilian
8	RAFAEL COLETTI	RODRIGO	Raphael	Wilian
9	KLAUS	RODRIGO	Elena	Sérgio
10	EDUARDO	SERGIO	Wilian	Rodrigo
11	MISAEL	SERGIO	Alexandre	Raphael
12	BRUNO	SERGIO	Elena	Regiane

Total de Bancas	
Elena	6
Regiane	5
Rodrigo	6
Sergio	7
Alexandre	4
Raphael	5

ORIENTADO	ORIENTADOR/COORIENTADOR	PROFESSOR 1	PROFESSOR 2	
11 DE JUNHO				TERÇA
	ACADÊMICO	ORIENTADOR	PROFESSOR 1	PROFESSOR 2
19h	FELIPE SOCZEK	REGIANE	ELENA	ALEXANDRE
20h	CAMILA	ELENA	SÉRGIO	RAPHAEL
21h	BENTO	REGIANE	RODRIGO	RAPHAEL
22h	RAFAEL COLETTI	RODRIGO	RAPHAEL	WILIAN
12 DE JUNHO				
	ACADÊMICO	ORIENTADOR	PROFESSOR 1	PROFESSOR 2
19h	BRUNO	SERGIO	ELENA	REGIANE
20h	GUSTAVO K.	RODRIGO	REGIANE	WILIAN
21h	ELOIR	REGIANE	SÉRGIO	ALEXANDRE
22h	MISAEL	SERGIO	ALEXANDRE	RAPHAEL
13 DE JUNHO				
	ACADÊMICO	ORIENTADOR	PROFESSOR 1	PROFESSOR 2
19h	CLAUDINEI	ELENA	SÉRGIO	ALEXANDRE
20h	SALLES	RODRIGO	ELENA	WILIAN
21h	KLAUS	RODRIGO	ELENA	SERGIO
22h	EDUARDO	SERGIO	WILIAN	RODRIGO